Supplementary Materials for 3D Point Cloud Generative Adversarial Network Based on Tree Structured Graph Convolutions

Dong Wook Shu[*], Sung Woo Park[*], and Junseok Kwon
School of Computer Science and Engineering, Chung-Ang University, Seoul, Korea
seowok@naver.com pswkiki@gmail.com jskwon@cau.ac.kr

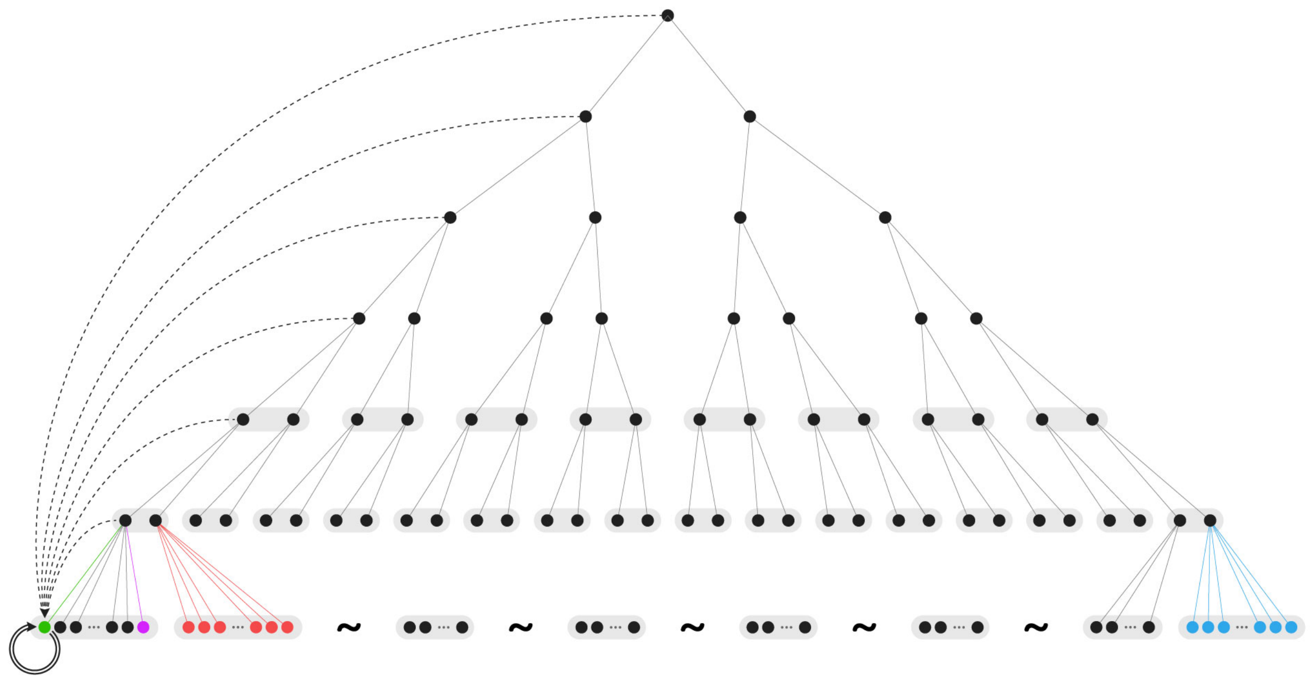

Figure A: **Genogram of the 3D point clouds in Fig. 6 of main paper.** The green and purple points have the same ancestors and split at the last layer. The red and blue point sets have entirely different ancestors except for the root.

.

1. Illustration of the Proposed Tree-structured Graph Convolution Network (treeGCN)

The proposed *loop* term with K supports is illustrated by double line arrows in Fig. A. Our loop term uses K parameters, while conventional GCNs [2] use only single parameter for the loop term. By using K parameters in loop term, our method enhances representation power for describing various typologies of 3D point clouds. This term is applied for each vertex of every layer after branching, as shown in (5) of the main paper.

The proposed *ancestor* term is illustrated by dotted line arrows in Fig. A. Conventional GCNs that use first-order approximation to extract the information from neighboring vertices have two problems for 3d point cloud generation. First, conventional GCNs even with a lot of layers have insufficient representation power for describing various 3d point clouds. The methods in [3, 4] attempted to solve the first problem by using fully-connected networks, but its representation power was still limited. Second, conventional GCNs require adjacency information as a prior knowledge, which is not available in

[*] Authors contributed equally

unsupervised 3d point clouds generation problems. The method in [4] used k-nearest neighbor techniques to solve the second problem, but it suffered from high computational complexity. The proposed ancestor term uses ancestor information within a tree, which boosts representation power and enables efficient computation. Our method uses ancestor vertices within a tree instead of neighbor vertices to extract features of the next layer, as shown in Fig. A. This term is applied for each vertex on every layer after branching, as shown in (6) of the main paper.
The proposed branching is illustrated by normal lines in Fig. A. Intuitively, we can know that the branching performs well on tree-like structures in Fig. A. There can be several different branching strategies, which are compared in the following section.

2. Ablation Study on Different Branching Strategies

As an ablation study on hyper-parameter settings, different branching strategies were compared. Fig. B shows the convergences of the FPD of our tree-GAN according to different branching strategies. As shown in Fig. B, the proposed tree-GAN is insensitive to different branching strategies, while each strategy is still useful for generating accurate point clouds in terms of the FPD. In experiments of the main paper, we used a branching degree $\{1, 2, 2, 2, 2, 2, 64\}$ for seven layers of the proposed generator (*i.e.*, treeGCN). For the discriminator, we used the same network proposed by [1]. However, in this experiment, we used a similar but larger discriminator $\{3, 64, 128, 256, 512, 1024\}$ than that of [1]. Note that different branching strategies do not significantly change convergence dynamics, but can affect point cloud distributions of semantic parts. Fig. C shows that rg degree of initial root determines output point distributions, which is consistent with mathematical properties of the proposed treeGCN (Section 5).

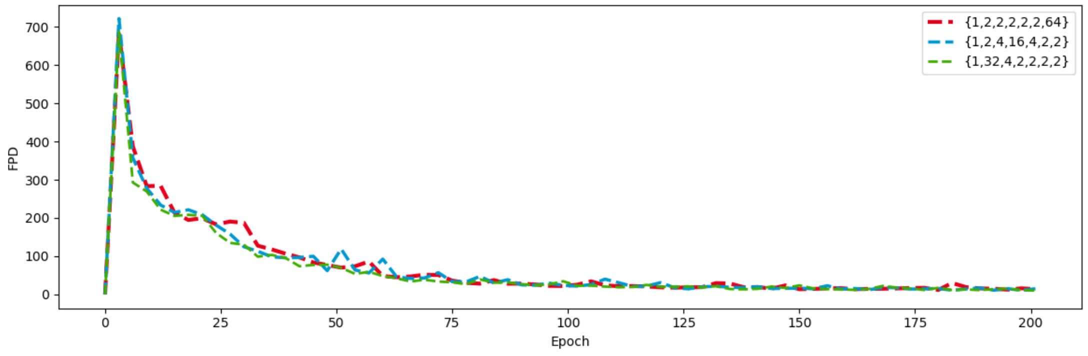

Figure B: **Convergence curves according to different branching strategies.** The branching strategies with different degrees $\{1, 2, 2, 2, 2, 2, 64\}$, $\{1, 2, 4, 16, 4, 2, 2\}$, and $\{1, 32, 4, 2, 2, 2, 2\}$ are denoted by red, blue, and green curves, respectively.

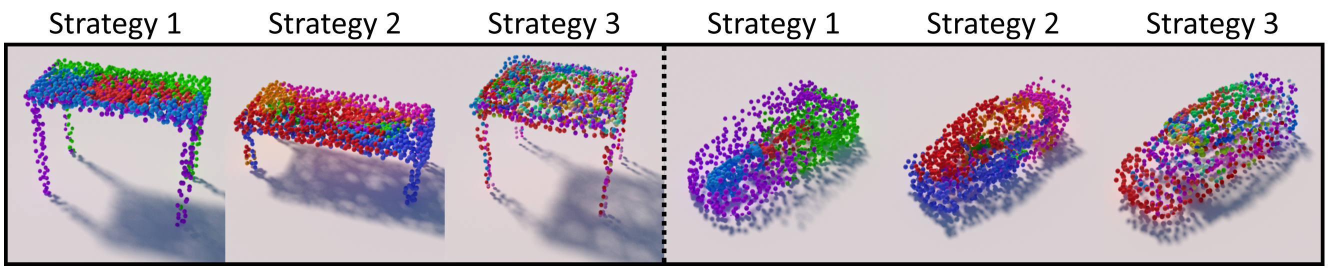

Figure C: **Unsupervised 3D point clouds generated by our tree-GAN with different branching strategies.** (*e.g.*, $\{1, 2, 2, 2, 2, 2, 64\}$, $\{1, 2, 4, 16, 4, 2, 2\}$, and $\{1, 32, 4, 2, 2, 2, 2\}$ from left to right). Different branching strategies affect output point distributions of semantic parts.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3D point clouds. In *ICLR*, 2018. 2

[2] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2017. 1

[3] Martin Simonovsky and Nikos Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *CVPR*, 2017. 1

[4] Diego Valsesia, Giulia Fracastoro, and Enrico Magli. Learning localized generative models for 3D point clouds via graph convolution. In *ICLR*, 2019. 1, 2